Engineering Optimization IV – Rodrigues et al. (Eds)
© 2015 Taylor & Francis Group, London, ISBN 978-1-138-02725-1

Optimization of the DLR SpaceLiner inside the integration environment RCE

S. Zur & A. Tröltzsch
Department of Simulation and Software Technology, German Aerospace Center (DLR), Cologne, Germany

ABSTRACT: In this paper, we would like to present the multidisciplinary analysis and optimization of the preliminary DLR SpaceLiner design concept using the distributed workflow-driven integration environment, called RCE (Remote Component Environment). As the considered disciplines together build a coupled system and a change of input in one simulation tool may affect the output of another one, MultiDisciplinary Optimization (MDO) techniques have to be applied. To be able to efficiently analyze and optimize the overall system of the SpaceLiner, the described software and simulation tools have been integrated as a process chain inside the integration environment RCE. RCE comes with a number of integrated ready-to-use algorithms, but more importantly, it is furthermore possible to integrate third-party and user-provided algorithms. The described optimization problem of minimizing the mass of the DLR SpaceLiner design concept is solved using several publicly available software codes which we compare in terms of quality of the solution and in terms of number of function evaluations. Most of the integrated engineering simulation tools do not provide derivatives of the objective and constraint functions such that only optimization methods which do approximate or do not need derivatives at all can be applied in our case.

1 INTRODUCTION

In today's world, travelling fast between two places got very common. For being even faster in the future, the DLR (German Aerospace Center) studies the concept of ultra fast passenger transport by means of combining aviation travel with space travel. The DLR SpaceLiner concept study (M. 2007) is designed to fly from Europe to Australia in only 90 minutes. Designing such a vehicle creates various challenges as one would expect. Many disciplines with their own expert knowledge (e.g. for aerodynamics, structure, thermal management) must be considered for realizing an optimal design. Every discipline has its own simulation codes and for optimizing the overall design one must be aware that the several disciplines are a coupled system. Multidisciplinary optimization (MDO) techniques are used to realize this coupling (Tröltzsch, Siggel, Kopp, & Schwanekamp 2014). In our case, for coupling the simulation codes of each discipline and apply an automatic optimization loop, the integration framework RCE (Remote Component Environment) will be used. This framework provides some advantages for the engineer. It is possible to integrate own simulation tools into the framework and couple them in a simple way. There is no need for the user to be concerned with the data flow as this is done by RCE in the background. A process chain with several simulation tools can be executed automatically (Seider, Basermann, Mischke, Siggel, Tröltzsch, & Zur 2013). Furthermore, RCE provides the possibility to optimize these process chains. Some optimization algorithms are already implemented in RCE, but it is possible to use self defined methods. For this reason, a Python-based interface has been developed. In Section 2 of this paper, we first introduce the disciplines that are involved in the optimization. Along with that, the simulation tools for the respective disciplines will be explained. The multidisciplinary optimization framework for the problem is also described. In Section 3 follows an overview of the integration framework RCE which was used to solve the problem, along with the optimization strategy. In Section 4, we will present some numerical results concerning the optimization of the SpaceLiner, followed by some concluding remarks in Section 5.

2 MULTIDISCIPLINARY DESIGN OPTIMIZATION

Preliminary design concepts are not just the expert knowlegde of one field of research. Since there are many disciplines involved, such as structure and mass optimization, many disciplines have to be coupled and will effect each other in the whole process. In this section, the disciplines for the preliminary SpaceLiner design concept as well as their efficent tools for the multidisciplinary simulation will be introduced.

2.1 *Involved disciplines and tools*

The first discipline needed for the optimization concerns the geometry. For finding an optimal design, the outer shape of the spacecraft must be parameterized. This obviously impacts many aspects of other disciplines such as the aerodynamic behavior or the stability. Changing the geometry in an optimization requires a changeable representation of it. This is done by generating a block-wise build up structured quadrilateral panel mesh with the help of the program GGH (Grid Generator for HOTSOSE), programmed by DLR-SART.

Following the geometry calculations, the next involved disciplines are the aero- and aerothermodynamics. The method HOTSOSE (HOT Second Order Shock Expansion) was implemented at DLR and estimates all aerodynamic coefficients as well as surface parameters and heat loads (Reisch & Anseaum 1998, Streit, Martin, & Eggers 1994).

The next discipline is the thermal protection system (TPS) involving transpiration cooling by water. The TPS is designed for protecting the structure, the internal systems and the passengers against the heat that is produced during the descent when re-entering the atmosphere.

The total mass of the spacecraft is influenced by the mass of the TPS. To consider this mass, the tool CalCoolAid calculates an estimation of the required cooling water mass based on the calculations of HOTSOSE.

After that, the mass model of the design can be approximated. This is done by the tool STSM (Space Transportation Systems Mass), which was also developed at DLR-SART.

The last discipline involved is the structural sizing. Using the analysis program HySAP (A. 2012), which was developed at DLR-SART, a rapid structural analysis on a preliminary design level for almost arbitrary vehicle configurations is calculated.

2.2 *Multidisciplinary design optimization problem*

So far, research has been carried out on the mentioned disciplines separately. The parameters of each major discipline have been optimized on its own with only little consideration of the other disciplines. As the considered disciplines together build a coupled system and a change of input in one simulation tool may affect the output of another one, multidisciplinary optimization (MDO) techniques have to be applied. The objective of the formulated MDO problem is to locate the design with the minimal mass including liquids subject to a lower bound on the glide ratio, an upper bound on the pitching moment and to several geometrical constraints.

In our case, the system is coupled with one back coupling. Note that values which are output to one discipline and input to another discipline are called coupling variables.

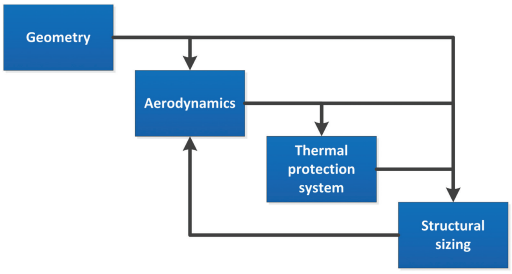


Figure 1. The coupled system of the disciplines for SpaceLiner flight analysis and optimization.

Several approaches have been developed to solve multidisciplinary optimization problems. Among others, Simultaneous Analysis and Design (SAND) (Haftka 1985), Individual Design Feasible (IDF) (Cramer, Dennis, Frank, Lewis, & Shubin 1994), Multidisciplinary Feasible (MDF) (Cramer, Dennis, Frank, Lewis, & Shubin 1994), Concurrent Subspace Optimization (CSSO) (Sobieszczanski-Sobieski 1988) and Collaborative Optimization (CO) (Braun 1996) are well-known methods.

We decided to use the "sequential Individual Design Feasible" (sIDF) approach which is a version of IDF. In IDF, the coupling variables are added to the set of design variables to decouple the discipline analyses so that they no longer rely on each other for their coupling variable input. To ensure a multidisciplinary feasible solution at the optimum, one additional feasibility constraint is added to the optimization problem for each coupling variable. These constraints ensure that at the optimum, the estimate of the coupling variables matches the actual coupling variables computed by each discipline. With f denoting the objective function and c denoting the constraint functions, IDF can be stated as,

$$\begin{aligned} \min \quad & f(z, x, y^t) \\ \text{w.r.t.} \quad & z, x, y^t \\ \text{s.t.} \quad & c(z, x, y(x, y^t, z)) \leq 0 \\ & y_i^t - y_i(x, y_j^t, z) = 0, \end{aligned} \tag{1}$$

where y^t represents the coupling variables estimates (or targets) provided by the optimizer, y_i are the coupling variable outputs of discipline i given the estimate of the non-local coupling variables y_j^t from discipline j. Furthermore, x represents the set of local design variables, which are only involved in one discipline and z are the global design variables which are involved in more than one discipline.

This architecture enables the discipline analyses to be performed in parallel, since the coupling between the disciplines is resolved by the coupling variable copies and consistency constraints. The advantage of

IDF compared to other approaches is that the IDF problem formulation is very compact and requires minimal modification to existing discipline analyses. Nevertheless, it is not recommended for problems with a large number of coupling variables.

The sIDF approach is a version of IDF where outputs of one discipline which are connected as inputs to only one other discipline are directly tied with the receiving discipline. In this case, coupling through the optimizer is avoided. This technique reduces the number of coupling variables and constraints which would be required in the pure IDF approach. On the other hand, the evaluation of the discipline outputs can not be fully performed in parallel anymore.

3 THE INTEGRATION ENVIRONMENT RCE

For finding an optimal solution considering all disciplines involved, the given simulation tools must be coupled together. Since the optimization loop should run automatically, the integration framework RCE (Remote Component Environment) is used to achieve this. RCE is an open source project which supports engineers to manage complex simulations in a collaborative environment. It is a software framework which allows easy creation and execution of process chains for coupling simulation codes in a graphical user interface. Furthermore, it provides a secure and uniform access of data also in a distributed environment where several engineering departments at different sites may be involved in the design process. As can be seen in Figure 2, where an integrated process chain of RCE is depicted, the tools and other parts of the process chain, such as the optimization algorithm, are displayed as boxes which are connected through arrows. These arrows visualize the data flow between the tools. If the process chain contains distributed tools in a network, these arrows also represent the transfer of the relevant data from one site to another without the engineers being concerned with it. All available tools in the network are listed in the graphical editor and can be used in the process chain editor by drag and drop. Each tool is only executed on the machine where it is integrated in RCE, such that for example dependencies to other libraries must only be configured on this machine.

3.1 *Integration of simulation tools*

In order to build up a process chain with its own simulation tools, this tools must be somehow integrated in RCE. This is not too complicated, as long as the tools fulfill the following requirements:

- The tool must be executable via command line calls, and without any user interaction during execution,
- the tool's input must only be command line parameters and files.

In the integration concept of RCE, an integrated tool is treated as a black box. I.e., it is seen in terms

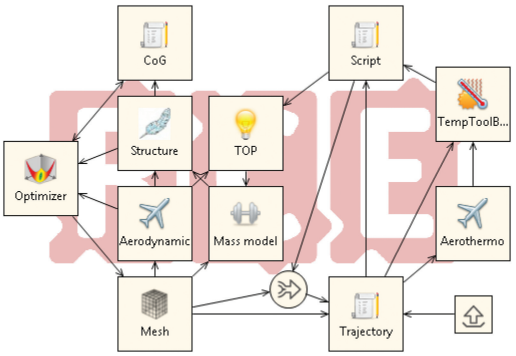


Figure 2. A process chain in RCE.

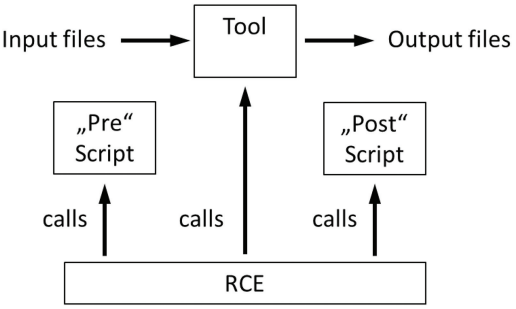


Figure 3. Tool integration concept of RCE.

of its inputs and outputs without any knowledge of its internal working. Figure 3 summarizes RCE's tool integration concept.

For integrating the tool, there is a step-by-step wizard that helps the engineer to configure everything that RCE needs for executing the tool, e.g. which inputs and outputs it has.

Once a tool has been integrated into one RCE instance, e.g., the tool developer's instance, it is instantly being distributed in the RCE network the correspondent machine is part of. As soon as another machine has received the information, this machine's user can immediately start using the tool in his process chains as if it is installed on its local machine. whereas the actual execution of the tool is always performed on the machine on which it has been integrated.

3.2 *Optimization in RCE*

RCE comes with an already integrated optimizer component. With this component, it is possible to define an optimization loop using the ready-to-use algorithms of the DAKOTA software toolkit from Sandia National Laboratories (Adams, Bauman, Bohnhoff, Ebeida, Eddy, Eldred, Hough, Hu, Jakeman, Swiler, & Vigil 2013). The toolkit provides algorithms for design optimization, uncertainty quantification, parameter estimation, design of experiments, and sensitivity analysis, and a range of parallel computing and simulation interfacing services.

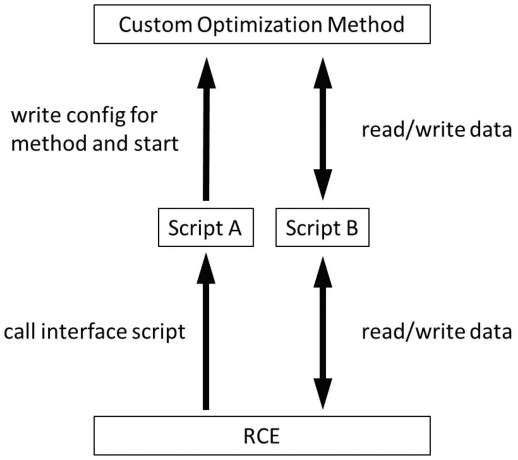


Figure 4. The optimizer interface in RCE.

But more importantly, it is possible to integrate and use own optimization algorithms and any other public domain or commercial optimization codes. This is achieved through an interface provided by RCE which is written in Python. The interface consists of two parts. The first part is the definition of parameters for the algorithm to be integrated. This can be parameters for stopping criteria, trust regions or gradient calculations for example. The definition of these parameters is important for being able to configure the new algorithm just with RCE's graphical user interface. The second part of the interface is the implementation of the algorithm itself. This is achieved using a given Python script that can be modified to fit the algorithm. It is also possible to use other optimization libraries or an in-house optimization algorithm with this interface. The parameters defined above and the definition of the design variables, objective functions and constraint functions are automatically written to configuration files by RCE and read by the interface. At last, there is one Python script which must not be changed, that manages the exchange of data between RCE and the optimization algorithm (see Figure 4).

Once a process chain that should be optimized is integrated in RCE as described above, several optimization algorithms can be used to optimize it. This does not only include the built-in algorithms, but every other optimization package. Once implemented in RCE, one can switch between the provided optimization methods very easily. Comparing several algorithms and finding the best one for the current process chain can be done very efficiently. Another advantage of the interface is that some algorithms that are generic can be adapted for the current problem.

4 RESULTS

The optimization of thermal management of the SpaceLiner uses RCE as platform to calculate the process chain. The different simulation tools mentioned in 2 are integrated using the in 3 described way. Since the involved experts for the disciplines are located on two different sites, RCE supports them to develop and integrate their own simulation codes and test it directly with the real world data process chain. This way, problems with the tools or even in the design of the process chain can be tracked down much faster. Using the built-in optimization algorithms, an optimized model of the SpaceLiner was calculated, shown in Figure 5.

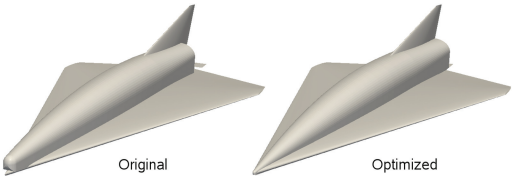


Figure 5. Optimization of the DLR SpaceLiner.

The focus is not only on optimizing the SpaceLiner itself, but also finding the best solver for it. Therefore, three solvers were tested. Two of them are from the RCE built-in DAKOTA library, which are COBYLA (Powell 1994) and APPS (Gray & Kolda 2006). The third one is SOLVOPT (Kuntsevich & Kappel 1997) from the Python Optimization Package pyOpt (Perez, Jansen, & Martins 2012), which was integrated using the Python interface described in 3.

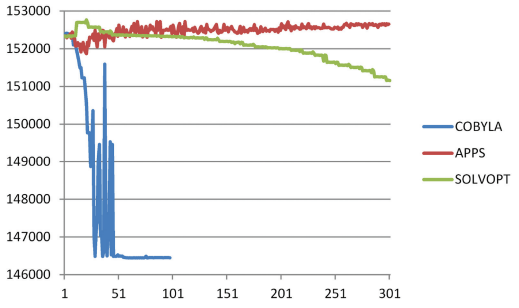
The numerical results are presented as histories in Figures 6(a) and 6(b) where all evaluated function values are displayed. The values of the objective function (mass of the spacecraft in kg) are depicted in 6(a) and the cumulated constraint violation in 6(b).

Regarding the objective function reduction in Figure 6(a), one could have the impression, that COBYLA is the best method for the problem. But the comparison of the constraint violation in Figure 6(b) shows that COBYLA does not satisfy the constraints at termination. Instead, we see that SOLVOPT is able to find a feasible solution inside the given time frame and that it is at the same time able to reduce the spacecraft mass for more than 1000 kg. The optimizer APPS is not able to provide a feasible solution nor a reduced objective function value in the given time frame.

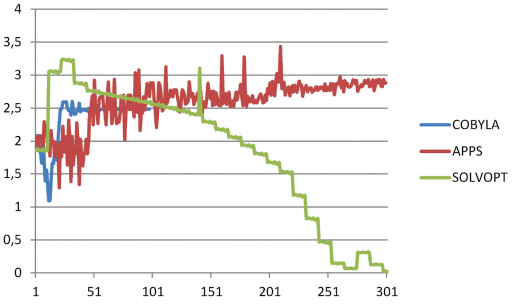
5 CONCLUSIONS

We presented the multidisciplinary analysis and optimization problem of the DLR SpaceLiner design concept involving the disciplines geometry, aerodynamics, thermal protection and structure. The goal was to find a feasible and possibly optimized solution of the whole system starting from the best solution of each single discipline. For this purpose, we implemented the given simulation tools of each discipline in the integration framework RCE using the sIDF approach.

We introduced a process chain, where we utilized the collaboration support provided by the integration framework RCE. In the presented use case of thermal



(a) Objective function



(b) Cumulated constraint violation

Figure 6. Comparison of COBYLA, APPS and SOLVOPT.

management in spacecraft design, RCE enables efficient collaboration between experts working at different sites. The tool specialists use newly implemented optimization methods immediately. The optimization specialists develop new methods always against the latest version of the involved tools. From an end user's point of view, the introduced approach allows time savings and simplifies collaborative work.

Based on experiences made in multidisciplinary optimization, RCE will be extended continuously. Currently, tools can be distributed from one expert to another. This does not ensure that an expert from another discipline knows how to use the tool correctly. RCE will address this issue in the future. Documentation of the tool will be directly integrated in RCE as well as an opportunity for instant messaging. A new visualization that focuses on multidisciplinary optimization is also planned.

Concerning the engineering simulation tools of the involved disciplines, several modifications and enhancement are planned in the near future.

Furthermore, to enhance the presented multidisciplinary analysis framework, we envisage to integrate an additional tool for a more reliable analysis of the passive thermal protection system of the SpaceLiner and another tool for the exact estimation of tank masses.

REFERENCES

A., K. (2012). Parametric studies for the structural predesign of hypersonic aerospace vehicles. In *12th European Conference on Spacecraft Structures, Materials and Environmental Testing, Noordwijk, The Netherlands*.

Adams, B., L. Bauman, K. Bohnhoff, W.J.and Dalbey, M. Ebeida, J. Eddy, M. Eldred, P. Hough, K. Hu, J. Jakeman, L. Swiler, & D. Vigil (December 2009. Updated April 2013). Dakota, a multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analysis: Version 5.4 user's manual. *Sandia Technical Report SAND2010-2183*.

Braun, R. D. (1996). Collaborative optimization: An architecture for large-scale distributed design. *Ph.D. thesis, Stanford University, Stanford, CA 94305*.

Cramer, E. J., J. J. E. Dennis, P. D. Frank, R. M. Lewis, & G. R. Shubin (1994). Problem formulation for multidisciplinary optimization. *SIAM Journal on Optimization, Vol. 4, No. 4, pp. 754–776*.

Gray, G. & T. Kolda (2006). Algorithm 856: Appspack 4.0: Asynchronous parallel pattern search for derivative-free optimization. *ACM Transactions on Mathematical Software, 32(3):485–507*.

Haftka, R. T. (1985). Simultaneous analysis and design. *AIAA Journal, Vol. 23, No. 7, pp. 1099–1103*.

Kuntsevich, A. & F. Kappel (1997). Solvopt manual: The solver for local nonlinear optimization problems. *Tech. rep., Institute for Mathematics, Karl Franzens University of Graz, Autria*.

M., S. (2007). Introducing the spaceliner vision. In *7th International Symposium on Launcher Technologies, Barcelona, Spain*.

Perez, R., P. Jansen, & J. Martins (2012). pyopt: A python-based object-oriented framework for nonlinear constrained optimization. *Structures and Multidisciplinary Optimization, 45(1):101–118*.

Powell, M. (1994). A direct search optimization method that models the objective and constraint functions by linear interpolation. *Advances in Optimization and Numerical Analysis, Kluwer Academic, Dordrecht, , pp 51–67*.

Reisch, U. & Y. Anseaum (1998). Validation of the approximate calculation procedure hotsose for aerodynamic and thermal loads in hypersonic flow with existing experimental and numerical results. *DLR-Forschungsbericht, 98–23*.

Seider, D., A. Basermann, R. Mischke, M. Siggel, A. Tröltzsch, & S. Zur (2013). Ad hoc collaborative design with focus on iterative multidisciplinary process chain development applied to thermal management of spacecraft. In *CEAS 2013, Linkoping, Sweden*.

Sobieszczanski-Sobieski, J. (1988). Optimization by decomposition: a step from hierarchic to non-hierarchic systems. *Tech. Rep. September, NASA Langley Research Center, Hampton, VA*.

Streit, T., S. Martin, & T. Eggers (1994). Approximate heat transfer methods for hypersonic flow in comparison with results provided by numerical navier-stokes solution. *DLR-Forschungsbericht, 94–36*.

Tröltzsch, A., M. Siggel, A. Kopp, & T. Schwanekamp (2014). Multidisciplinary analysis of the dlr spaceliner concept by different optimization techniques. In *Minisymposium on structural and multidisciplinary optimization at the World Congress on Computational Mechanics 2014, Barcelona, Spain*.